計算の手間の評価とプログラムの動的解析システム (計算の手間とデータ構造)

AUTHOR(S):

牛島, 和夫; 藤村, 直美

CITATION:

牛島, 和夫 ...[et al]. 計算の手間の評価とプログラムの動的解析システム (計算の手間とデータ構造). 数理解析研究所講究録 1975, 250: 163-180

ISSUE DATE:

1975-09

URL:

http://hdl.handle.net/2433/105698

数理解析研究所講究録
250巻 1975年 163-180

計算の手間の評価とプログラムの動的解析システム

九州大学工学部　牛島和夫

藤村直美

1．FORDAP システム

プログラムは，まず理解しやすいことが大切である。その上で効率の良さが要求される。ところが，自分の作成したプログラムの実際の動作をプログラマが客観的に認識するのはなかなか容易でない。プログラムの実行によつて処理系から得られるプログラムの動的特性に関する情報は，プログラムが異常終了した場合は，そのエラーコードとアドレス，正常終了の場合は，実行全体に要した時間というのが普通であろう。この上さらにプログラム中の各実行文毎に実行回数や実行に要した時間に関する情報が得られれば，プログラムの改善に役立たせることができる。そこでプログラムの動的特性を測定し，上述のような情報をプログラマに返す目的をもつたFORDAP[1]（FORTUNE[2]）と呼ばれるシステムを，若干の仕様変更を行なつてFACOM230-45Sで実用化した。以下これをFORDAPと称する。

FORDAPは主としてつぎのような機能をもつている。

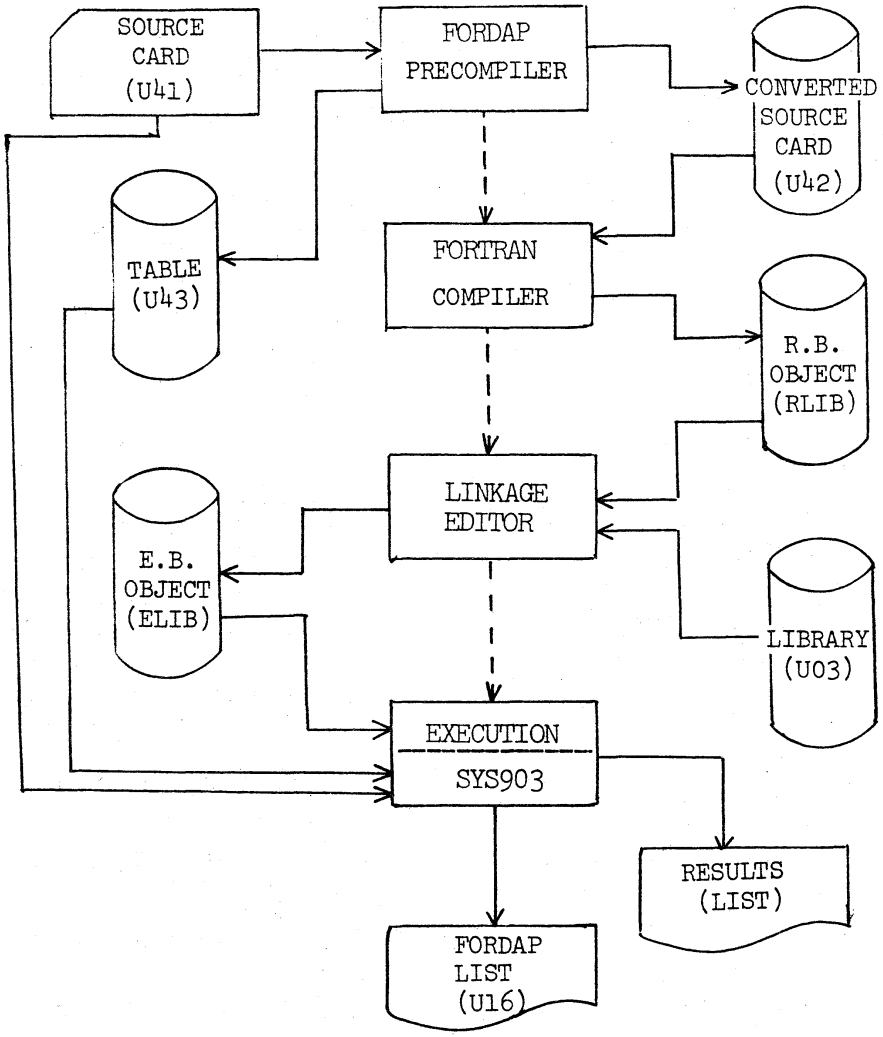

図1　FORDAP システムの大略の流れ

(1) FORTRANプログラムの各実行文の実行回数（論理ＩＦ文にあつては，さらに論理式の値が真となる回数）を計数する。

(2)プログラム単位毎に実際に要したCPU timeを計測し実行のコストとする。（計測によるオーバヘッドが含まれる）

(3)デイバツグ支援のため，１６通りの異常終了に対してそれまでに計測した (1)～(2)の情報と計算結果を出力する。

```
      SOURCE STATEMENT          7261 MSEC EXECUTIONS     TRUE

      INTEGER*4 M,M1,M2
      COMMON A(13000)/SUM/S,N
      DO 100 J=3,3                                          1
      N=25*2**J                                             1
      WRITE(6,699)                                          1
  699 FORMAT(1H )
      DO 100 I=1,2                                          1
      CALL ASGN(1)                                          2
      CALL CLOCKM(M1)                                       2
      GO TO (31,32),I                                       2
   31 CONTINUE                                              1
      CALL SUM301                                           1
      GO TO 99                                              1
   32 CONTINUE                                              1
      CALL SUM201                                           1
      GO TO 99                                              1
   99 CALL CLOCKM(M2)                                       2
      M=M2-M1                                               2
      WRITE(6,600) N,I,M,S                                  2
  100 CONTINUE                                              2
      STOP                                                  1
  600 FORMAT(     略          )
      END
      SOURCE STATEMENT          1562 MSEC EXECUTIONS     TRUE

      SUBROUTINE SUM201                                     1
      COMMON A(2100)
      COMMON/SUM/S,NN
      N=NN                                                  1
    1 II=1                                                199
      IF(A(1).GT.A(2))II=2                                199    120 ( 60.3% )
      III=3-II                                            199
      IF(N.EQ.2) GO TO 3                                  199      1 (  0.5% )
      DO 2 I=3,N                                          198
      IF(A(I).GE.A(III)) GO TO 2                        19701  18484 ( 93.8% )
      III =I                                             1217
      IF(A(I).GE.A(II)) GO TO 2                          1217    606 ( 49.8% )
      III=II                                              611
      II =I                                               611
    2 CONTINUE                                          19701
    3 S=A(II)+A(III)                                      199
      IF(N.EQ.2) RETURN                                   199      1 (  0.5% )
      A(I)=S                                              198
      A(III)=A(N)                                         198
      N=N-1                                               198
      GO TO 1                                             198
      END

      SOURCE STATEMENT          5655 MSEC EXECUTIONS     TRUE

      SUBROUTINE SUM301                                     1
      COMMON A(2100)
      COMMON/SUM/S,N
      DO 30 J=2,N                                           1
   20 ISW=0                                               584
      DO 10 I=J,N                                         584
      IF(A(I-1).LE.A(I)) GO TO 10                       76817  53337 ( 69.4% )
      W=A(I)                                            23480
      A(I)=A(I-1)                                       23480
      A(I-1)=W                                          23480
      ISW=1                                             23480
   10 CONTINUE                                          76817
      IF(ISW.NE.0) GO TO 20                               584    385 ( 65.9% )
   30 A(J)=A(J)+A(J-1)                                    199
      S=A(N)                                                1
      RETURN                                                1
      END
```

図 2　出力の例

図１にシステムの大略の流れを示した。図２は，システムによる出力例である。このリストの後にこのプログラムの実行による結果の印刷がつづく。

２．FORDAPを利用したプログラムの段階的改善

<u>問題</u>　ｎ個(≧2)の正の浮動小数点数A(1),...,A(n) から最小の数と２番目に小さい数を見つけて加え，次にその残りに結果を含めたｎ－１個の正の浮動小数点数の中から再び小さい方から２つの数を見つけて加える。以下同様の操作をくり返して，最終的にｎ個の正の浮動小数点数の総和を求めるプログラムを作る。（文献(4)，(5)でHuffman 法と呼んでいるもの。）

つぎのような２通りの考え方でプログラムしてみる。

(a) ｍ個の中から最小の数の場所ｋ，２番目の数の場所 l を探して A(k) = A(k) + A(l), A(l) = A(m) をm=n, n-1,..., 2 について行なう方法。

(b) n-j+1個のデータをまず小さい順に分類し (A(j) ≦ A(j+1) ≦ ...≦ A(n)), A(j+1) = A(j) + A(j+1) をつくる。これをｊ = 1, 2,...,n－1についてくり返せばA(n)に総和を得る。

(a)，(b)の方法を文の数を比較的少なくして実現したものがそれぞれ SUBROUTINE SUM201,および SUM301 である。（図２）　n = 200 として A(1),..., A(n)には，(0, 1) の一様乱数を ASGN によつて与え

```
   SOURCE STATEMENT          3130 MSEC EXECUTIONS     TRUE

      SUBROUTINE SUM302                                    1
      COMMON A(2000)/SUM/S,N
      CALL QUICKS                                          1
      A(2)=A(2)+A(1)                                       1
      DO 30 J=3,N                                          1
20      ISW=0                                            394
        DO 10 I=J,N                                      394
          IF(A(I-1).LE.A(I)) GO TO 10                  39007  25519 ( 65.4% )
            W=A(I)                                     13488
            A(I)=A(I-1)                                13488
            A(I-1)=W                                   13488
            ISW=1                                      13488
10      CONTINUE                                       39007
        IF(ISW.NE.0) GO TO 20                            394    196 ( 49.7% )
30    A(J)=A(J)+A(J-1)                                   198
      S=A(N)                                               1
      RETURN                                               1
      END


   SOURCE STATEMENT           157 MSEC EXECUTIONS     TRUE

      SUBROUTINE QUICKS                                    1
      COMMON A(2000)
      COMMON/SUM/S,N
      INTEGER STACK(40),LOWER,UPPER,MIDDLE,P
      A(N+1)=1.E30
      LOWER=1                                              1
      UPPER=N                                              1
      P=1                                                  1
 1    IF(LOWER.GE.UPPER) GO TO 2                         275    138 ( 50.2% )
        CALL PARTN(LOWER,UPPER,MIDDLE)                   137
        STACK(P)=MIDDLE+1                                137
        STACK(P+1)=UPPER                                 137
        P=P+2                                            137
        UPPER=MIDDLE-1                                   137
        GO TO 1                                          137
 2    IF(P.LE.1) RETURN                                  138      1 (  0.7% )
        P=P-2                                            137
        LOWER=STACK(P)                                   137
        UPPER=STACK(P+1)                                 137
        GO TO 1                                          137
      END

   SOURCE STATEMENT           128 MSEC EXECUTIONS     TRUE

      SUBROUTINE PARTN(LOWER,UPPER,MIDDLE)               137
      COMMON A(2100)
      INTEGER LOWER,UPPER,MIDDLE
      M=UPPER+1                                        137
      L=LOWER                                          137
      W=A(L)                                           137
10    M=M-1                                            927
      IF(A(M).GT.W) GO TO 10                           927    544 ( 58.7% )
        A(L)=A(M)                                      383
20    L=L+1                                            774
      IF(A(L).LT.W) GO TO 20                           774    391 ( 50.5% )
        IF(L.GE.M) GO TO 30                            383    137 ( 35.8% )
          A(M)=A(L)                                    246
      GO TO 10                                         246
30    IF(L.NE.M) L=L-1                                 137    137 (100.0% )
        A(L)=W                                         137
        MIDDLE=L                                       137
      RETURN                                           137
      END
```

図 3

る。FORDAP システムのもとで実行させてみると図 2 に示すように

SUM201 1562msec, SUM301 5655msec

となり，201 が断然優れている。とにかく301を201程度まで速くすることを追求してみる。301では，DO 10 I = ...のループに実行回数が集中しているのでこれを減少させることを考えよう。301はjについて毎回泡立ち法で分類を行なつている。この方法の手間はn^2に比例することが知られているので，これをn log(n)に比例する方法（例えば heap sort, quick sort など）に取りかえることが考えられる。ところで j = 2 以後はデータはほとんど整列しており，とにかく j = 1 のときだけを quick sort(SUBROUTINE QUICKS, 図 3 b, c）に切りかえてみた（SUM302, 図 3.a).

SUM302 3130msec （うち QUICKS 157msec）

ここで A(I-1)と A(I)の比較が 76817 回→ 39007 回，A(I-1), A(I) の交換が 23480回 →13488 回に減少した。これらをさらに減少させることを試みる。例えば j = 2 のとき列を乱しているのはA(2)のみである。したがつてA(2)のおさまる場所が泡立ち法で決まれば，あとはつねに A(I-1) ≦ A(I) となるので，泡立ち法でデータの交換があつたことを示すフラグISW は不要であることに気付く。ISW に関連する文を除いたものが SUM303(図 4.a）である。

SUM303 1736msec

A(I-1), A(I)の比較 19701 回，交換は302と同じ。

```
      CALL QUICKS                                          1
      A(2)=A(2)+A(1)                                       1
      DO 30 J=3,N                                          1
        DO 10 I=J,N                                      198
          IF(A(I-1).LE.A(I)) GO TO 10                  19701  6213 (  31.5% )
            W=A(I)                                     13488
            A(I)=A(I-1)                                13488
            A(I-1)=W                                   13488
10      CONTINUE                                       19701
30    A(J)=A(J)+A(J-1)                                   198
      S=A(N)                                               1
```

(a) SUM303

```
      DO 30 J=3,N                                          1
        DO 10 I=J,N                                      198
          IF(A(I-1).LE.A(I)) GO TO 30                  13549    61 (   0.5% )
            W=A(I)                                     13488
            A(I)=A(I-1)                                13488
            A(I-1)=W                                   13488
10      CONTINUE                                       13488
30    A(J)=A(J)+A(J-1)                                   198
```

(b) SUM304

図 4

201の時間とほゞ対等になつてきた。303 では，一たんおさまる場所がきまれば，あとは A(I-1) ≦ A(I) であるのに，構わずこの比較を続けている。その回数が6213回ということになる。したがつて，一たん A(I-1) ≦ A(I) が成立するやたゞちにループの外に脱出してよい。すなわち GOTO 10をGOTO 30に書きかえる。(SUM304，図4.b)

SUM304 1324msec ，比較回数 13549回

ここではじめて 201を抜くことができた。これをもつと短縮するためにA(I)とA(I-1)の交換回数を減らすことを考えた。 301→ 304 の段階的改善の過程をすべて示したが，この道すじは廻り道であつたようである。304 をみると A(I-1) > A(I) のとき両者の交換が行なわれ，さらに A(I) > A(I+1) ならば，再び交換が行なわれる。このように順送りにA(J)の落ちつく場所が決まるようになつている。しかし A(J+1),...,A(n) は整列しているのであるから

(a) A(I-1) ≦ A(J) < A(I)となるIを探して

(b)A(J+1),...,A(I-1)を1つずつ前にずらしてA(I-1) = A(J) とすればよい。

(a)をSUBROUTINE SEARCH(M, N, X, I): A(M) ≦ ...≦ A(N) の順に並んだデータ列の間にA(I-1) ≦ X < A(I) をみたすIをみつけて，引数Iに返す。A(M) → A(N) の一方向探索による。（図5.b)

(b)をSUBROUTINE INSERT(M, N, S): A(M), ..., A(N) を1つずつ前にずらしてA(N)=S とする。（図5.c)このようにモジュール化した。これらを用いたのがSUM305 （図5.a) である。

SUM305 1503msec(うち QUICKS 157msec, SEARCH 714msec, INSERT 595msec)

かえつて 304(1324msec)より増加してしまつた。これは1つは，SEARCH, INSERT をモジュール化したためにこれらが198回呼びだされ，引数などの結合の時間が無視できなくなつたこと，第2にFORDAP システムによつているため，サブルーチンの入口と出口に時間計測ルーチン (78.7 μsec／回）が挿入されており，約78.7 × 198 × 10^{-3}msec が加わつているためである。そこで FORDAP 抜きの時間も測定してみる。その前に305で所要時間の5割弱占めているSEARCH を二分探索法に変えたもの(SUBROUTINE SEARCB(M,N,X,I), 図5,d)を用いたSUM306では，SEARCH → SEARCB が714msec → 208msec となつて

SUM306 990msec

```
   SOURCE STATEMENT          1510 MSEC EXECUTIONS  TRUE

   SUBROUTINE SUM305                                    1
   COMMON A(2100)
   COMMON/SUM/S,N
   CALL QUICKS                                          1
   J=2                                                  1
10 S=A(J)+A(J-1)                                      199
   IF(J.EQ.N) RETURN                                  199    1 (  0.5% )
     CALL SEARCH(J+1,N,S,L)                           198
     CALL INSERT(J,L-1,S)                             198
     J=J+1                                            198
   GO TO 10                                           198
   END
                  (a) SUM305

   SOURCE STATEMENT          714 MSEC EXECUTIONS   TRUE

   SUBROUTINE SEARCH(M,N,X,I)                         198
   COMMON A(2100)
   A(N+1)=1.E30
   J=M                                                198
   GO TO 2                                            198
 1 J=J+2                                             6696
 2 IF(A(J).GT.X) GO TO 4                             6894  101 (  1.5% )
   IF(A(J+1).GT.X) GO TO 3                           6793   97 (  1.4% )
   GO TO 1                                           6696
 3 J=J+1                                               97
 4 I=J                                                198
   RETURN                                             198
   END
                  (b) SEARCH

   SOURCE STATEMENT          595 MSEC EXECUTIONS   TRUE

   SUBROUTINE INSERT(M,N,S)                           198
   COMMON A(2100)
   IF(M.EQ.N) GO TO 2                                 198    1 (  0.5% )
   N1=N-1                                             197
   DO 1 I=M,N1                                        197
 1 A(I)=A(I+1)                                      13489
 2 A(N)=S                                             198
   RETURN                                             198
   END
                  (c) INSERT

   SOURCE STATEMENT          208 MSEC EXECUTIONS   TRUE

   SUBROUTINE SEARCB(M,N,X,I)                         198
   COMMON A(2100)
   K=M-1                                              198
   I=N+1                                              198
 1 IF(I-K.EQ.1) RETURN                               1496   198 ( 13.2% )
   L=(I+K)/2                                          1298
   IF(A(L).GT.X) I=L                                  1298   243 ( 18.7% )
   IF(A(L).LE.X) K=L                                  1298  1055 ( 81.3% )
   GO TO 1                                            1298
   END
                  (d) SEARCB
```

図 5

となり，著るしい改善が見られた。ここで FORDAP をかけない本来の実行時間（これを実時間と称する。また FORDAP による時間をF時間と呼ぶ。）を比較したものが表1である。この表で

(1) 303 → 304 でF時間では減少しているが，実時間ではかえつて増加している。これは，304の GOTO 30 がループの外への脱出であるため，A(I) などの添字に対するコンパイラの最適化効果を妨げているためと思われる。303 は GOTO 10 がループ内への飛越であり，添字の最適化効果が有効に働いているためと思われる。

(2) 304 → 305　実時間では改善が著るしい。

(3) 306で最も時間を要しているのは INSERT であつた。これを減じるために FORTRAN 語で工夫もありうるが，参考のために FACOM 230-45S の MOVE 命令（256バイトまでを1命令でシフトできる）を用いて INSERT をアセンブリ語で書直したものによるものが表1 307 である。なおF時間で (448) は，INSERT には FORDAP の影響がないことを示す。

アルゴリズムの見直し　ここまで，FORDAP による実行回数やF時間をたよりに，実行の重みの大きい部分をつぶすというやり方で時間の短縮をはかつてきた。306では，SEARCB と INSERT は n－2回呼び出されている。サブルーチンの時間短縮には限界がみえているので，このn－2回を減らすことが可能かどうかを考える。

あるjについて，S = A(j) + A(j + 1) とし，A(i-1) ≦ S < A(i)

方法＼n	1	2	4	8	16	2 (Ftime)
201	135	518	2016	7906	31304	1562
301	391	1638	6527	26142	106340	5655
302	247	932	3608	14125	56106	3130
303	156	576	2170	8363	33010	1736
304	174	643	2382	9088	36151	1324
305	138	472	1672	6198	24237	1503
306	113	320	967	3140	11142	990
307	86	214	552	1521	4466	(448)

$\times 10^2$

表１．
各方法の実時間の比較(1)
（単位 : msec）
FACOM 230-45S
FORTRAN S(OPT)

となる最初のiが見つかれば，A(i)の手前の要素について

(i ≧ j + k, k = 0, 1, 2, ...)

S = A(j) + A(j+1) ≦ A(j+2) + A(j+3) ≦ A(j+4) + A(j+5) ≦ ...

が成立する。そこで改めて

(a) A(i) の手前の要素について

A(1) = A(j) + A(j+1), A(2) = A(j+2) + A(j+3),...

なる系列を作つて

(b)系列 A(i) ≦ A(i + 1) ≦ ... ≦ A(n) とまぜ合せを行なえばよいことになる。したがつて一度iがみつかると一時に (i-j)/2 個の和を作れるのでループの回数はかなり減ずることが期待できる。（ただしこれはj－iが偶数の場合。奇数の場合は系列A(i-1), A(i),...とまぜ合せることになる。）

(a)の処理のために，SUBROUTINE TSUM(M, A, B) (図6.b),

(b)の処理のために，SUBROUTINE MERGE(A, M, B, N, C):

それぞれ小→大順に並んだ2つの系列 A(1), ..., A(M) とB(1),...,

174

```
    SOURCE STATEMENT          2825 MSEC EXECUTIONS   TRUE

    SUBROUTINE SUM308                                    1
    COMMON A(2100)
    COMMON/SUM/S,N
    CALL QUICKS                                          1
    J=2                                                  1
10  S=A(J)+A(J-1)                                       20
    IF(J.EQ.N)RETURN                                    20    1 (  5.0% )
    CALL SEARCB(J+1,N,S,M)                              19
      K=(M-J+1)/2                                       19
    CALL TSUM(K,A(1),A(J-1))                            19
      JK=J-1+K                                          19
      JKK=JK+K                                          19
    CALL MERGE(A(1),K,A(JKK),N+1-JKK,A(JK))             19
    J=J+K                                               19
    GO TO 10                                            19
    END
```

(a) SUM308

```
    SOURCE STATEMENT            86 MSEC EXECUTIONS   TRUE

    SUBROUTINE TSUM(M,A,B)                              19
    DIMENSION A(2),B(2)
      DO 1 I=1,M                                        19
1     A(I)=B(2*I-1)+B(2*I)                            1598
    RETURN                                              19
    END
```

(b) TSUM

```
    SOURCE STATEMENT           997 MSEC EXECUTIONS   TRUE

    SUBROUTINE MERGE(A,M,B,N,C)                         19
    DIMENSION A(2),B(2),C(2)
    I=1                                                 19
    J=1                                                 19
    K=1                                                 19
10  IF(A(I).LT.B(J)) GO TO 20                         2431   860 ( 35.4% )
      C(K)=B(J)                                       1571
      J=J+1                                           1571
      K=K+1                                           1571
      IF(J.LE.N) GO TO 10                             1571  1566 ( 99.7% )
        DO 1 L=I,M                                       5
        KL=K+L-I                                       738
1       C(KL)=A(L)                                     738
      RETURN                                             5
20    C(K)=A(I)                                        860
      I=I+1                                            860
      K=K+1                                            860
      IF(I.LE.M) GO TO 10                              860   846 ( 98.4% )
        DO 2 L=J,N                                      14
        KL=K+L-J                                     12516
2       C(KL)=B(L)                                   12516
      RETURN                                            14
    END
```

(c) MERGE

図 6

B(N)をまぜ合せてC(1), ..., C(M+N) を作る（図 6.c) を用意する。これらを用いたプログラムをSUM308（図 6. a）としてn = 200, 1600 について FORDAP にかけてみた。

n	Total	QUICKS(PARTN)	SEARCB	TSUM	MERGE	ループの回数
200	280	157(128)	12	11	92	13
1600	2825	1708(1486)	25	86	997	19

QUICKS は Knuth[6] を参考にしてさらに FORDAP の助けを借りすでにかなり高速化をはかつたものである。ここではMERGE の減少を考える。308 が MERGE を引用しているのはたゞ１ケ所

```
CALL MERGE(A(1), K, A(JKK), N+1-JKK, A(JK))
```

であり，非常に特殊な結合である。一方図6.cでDO 2... の３行のループが 12516回と他に比べて著るしい。この部分に上の実引数を具体的にあてはめてみると，実は A(L) = A(L)という代入がくり返されていることがわかる。MERGEは一般的なまぜ合せのプログラムであつたが，このように問題の特殊性を利用すれば上の３行をMERGEから削除してよい。（これを SUBROUTINE MERGES とする）これを引用するプログラムをSUM309とするとこのF時間は，つぎのように著るしく改善された。

n	Total	QUICKS(PARTN)	SEARCB	TSUM	MERGES	ループの回数
200	227	157(128)	12	11	41	13
1600	2147	1714(1490)	25	87	311	19

さて n = 200のときループの数は 198回から 13 回に，n = 1600で1598回から 19 回に減じた。実際に各ループ毎に，一時に計算できる和の組数（プログラム中 K = (M-J+1)/2 で与えられる）を検討してみる。 MERGES の実引数 K ≧ 1は保証されているが，JKK(=JK+K=J-1+K*2)の値がN＋1になることがありうる。このことは，系列 A(1), A(2), ..., A(K) に対する相手の系列の長さが0ということを意味し，まぜ合せは不要である。したがつて MERGES は呼ばずにNの値をKで置きかえるだけでよい。このプログラムを SUM310（図7）とし，同じデータについて， FORDAP で計算するとつぎのようになつた。

n	Total	QUICKS(PARTN)	SEARCB	TSUM	MERGES	ループの回数（JKK=N+1となる回数）
200	213	157(128)	12	11	28	13 (5)
1600	2079	1690(1470)	25	85	270	19 (5)

MERGES についてだけみれば，これによる改善は1割強（n = 1600で 311msec → 270msec）であるが，全体に対する寄与は，QUICKS が全体の8割程度を占めているので，軽微である。表2に 308, 309, 310, QS(FOR)(QUICKS の処理時間），102(A(1), ..., A(n) を倍精度変数に単純に加えて，最後に単精度に丸める方法）の FACOM230-45S による実時間を計測したものを示した。nの大きいところでは，QUICKS が実時間で 310 の場合全体の 80% 以上を占めている。そこで参考のため，QUICKS が多数回呼んでいる PARTN をアセンブリ語で書き直したものによる時間 QS(ASS)も付加えた。このようにすれば

```
SO MCP STATEME T          2.79 MSEC EXECUTIONS  TRACE

      SUBROUTINE SUM310                                          1
      COMMON A(2000)/SUM/S,M
      N=M                                                        1
      CALL CLOCKS                                                1
      COMMON E                                                   6
      I=2                                                        6
      A(M+1)=1.E30                                               6
    1 S=A(J)+A(J-1)                                             20
      IF(J.GE.M) RETURN                                         20    1 (  5.0% )
      CALL SEARCH(J+1,M,S,M)                                    19
      K=(M-J+1)/2                                               19
      CALL TSUM(K,A(1),A(J-1))                                  19
      JK=J-1+K                                                  19
      JKK=JK+K                                                  19
      IF(JKK.EQ.N+1) GO TO 20                                   19    5 ( 26.3% )
      CALL MERGES(A(1),K,A(JKK),I-JKK,A(JK))                    14
      J=J+K                                                     14
      GO TO 1                                                   14
    2 K=K                                                        5
      GO TO 1                                                    5
      END
```

図 7

310 で n = 1600 のとき 1844 - 1513 + 808 = 1139msec で QS(ASS) の占める割合は約 70% まで縮小できる。表1，2は，1通りのデータに対するもので統計処理を施していない点に多少問題があろうが，プログラムの改善を説明する目的は，これで十分達していると思われる。図8に表1，2の主なものをまとめてみた。ここで310'などは，310 から QS(FOR) の時間を差引いた時間を意味している。この図から改善の様子（n^2からn log(n)以下に）をはつきり認めることができる。なお 308~310 で TSUM や MERGE(S) の実引数として共通ブロック内のデータを用いている。この結合は，JIS FORTRAN 8.4.2 の規定に触れるおそれがないわけではないが，ここでの目的を達しているのでこのまゝにとどめておくことにしたい。

方法＼n	1	2	4	8	16	32	64	128
308	51	119	267	567	1236	2640	5568	11585
309	44	93	200	420	878	1872	4012	8266
310	42	88	193	405	861	1844	3915	7921
QS(FOR)	28	63	146	320	693	1513	3291	6745
QS(ASS)	16	36	81	175	375	808	1739	3569
102	3	4	9	18	35	70	141	283

$\times 10^2$

表２．各方法の実時間の比較(2)（単位：msec）

FACOM230-45S FORTRAN S(OPT)

３．考察

３．１ FORDAP の使用経験　２節の段階的改善の解説中で具体的な問題にはその都度ふれた。このほか時間計測の便宜がプログラムのモジュール化を促進すること，異常終了点の発見と原因の究明の容易さ，一度も実行されていない文を直ちに発見できることなどがあげられる。

３．２ 実時間とF時間の関係　２節の例題では１：２～３，FORDAP のプリコンパイラ（FORTRAN で書かれている）では１：1.2 程度である。

３．３ 計算の手間と処理系依存性　コンパイラの最適化度やハードウエアの違い。

なお，東工大木村研究室においても，FORTUNE（我々の FORDAP とは若干仕様が異なる）が実用化させている。FORDAP 作成に際し

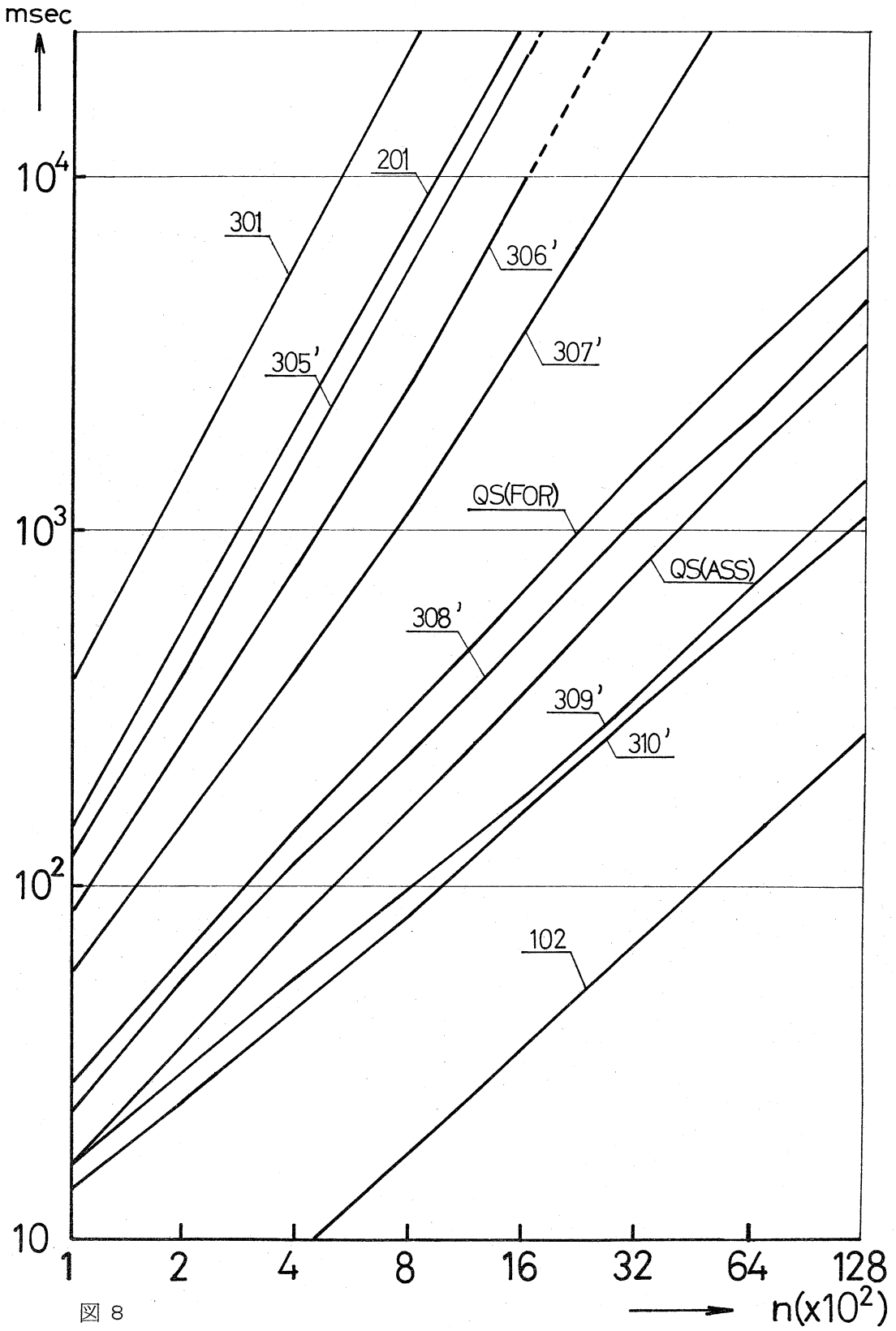
msec
10^4
10^3
10^2
10
201
301
306'
305'
307'
QS(FOR)
QS(ASS)
308'
309'
310'
102
1
2
4
8
16
32
64
128
n(x10²)

図 8

て種々議論をいただいた木村泉助教授に謝意を表わす。

参考文献

1. D.E. Knuth: An Empirical Study of FORTRAN Programs, Software, Vol.1(1971) pp.105-133.

2. D. Ingalls: The Execution Time Profile as a Programming Tool, R. Rustin ed., Design and Optimization of Compilers, Prentice Hall, 1972, pp.107-128.

3. 藤村，牛島：プログラムの実行解析システムの作成と使用について，九大工学集報 Vol.48, No.4 （印刷中）.

4. 牛島，芦田：浮動小数点数の総和の計算法の比較，数理解析研究所講究録 215 (1974 年 7月) pp.75-86.

5. O. Caprani: Round off Errors in Floating-Point Summation, BIT Vol.15, No.1(1975), pp.5-9.

6. D.E. Knuth: Structured Programming with go to Statements, Computing Survey, Vol.6(1974) pp.261-301.